Algorithmics

Sebastian Iwanowski
FH Wedel

5. String Matching

Algorithmics 5

String Matching

Task: Given a text $T = \{t_1,\ldots,t_n\}$ with n literals and a pattern $P = \{p_1,\ldots,p_m\}$ with m literals: Find the starting positions where P occurs in T.

naive algorithm: needs $O(nm)$ time

Algorithm of Knuth-Morris-Pratt: needs $O(n)$ time

Def.: P_q denotes the prefix of P consisting of the first q literals.

Def.: The prefix function $\pi: \mathbb{N}\setminus\{0\} \rightarrow \mathbb{N}$ for the pattern P is defined as:
$\pi(q) = k \Leftrightarrow$ k is the length of the longest strict prefix of P_q (*strict* means: $k < q$) which is also a Suffix of P_q

General method of the KMP algorithm:

For each $q \leq m$, compute the value $\pi(q)$ of the prefix function and store it.
Then scan T in only one iteration and shift P at any mismatch in pattern position q by $q - \pi(q)$.

In class: Why is this correct?

References:

Alt, Kap. 4.8
Cormen, ch. 32 (String matching), esp. 32.4 (KMP)

Algorithmics 5

String Matching

Algorithm of Knuth-Morris-Pratt:

needs O(n) time

Implementation of main procedure (version of Cormen):

```
i := 1; q := 0;
while i ≤ n do
{
    while (q>0)and (T[i] ≠ P[q+1])
        q := π (q);
    if T[i] = P[q+1] then q := q+1;
    if q = m
        then
        {
            print („Matching at position “, i-m);
            q := π (q);
        }
    i := i+1;
}
```

Invariant: q corresponds to an index such that (T[i-q+1],…,T[i]) coincides with (P[1],…,P[q])

To be considered with this version:
Why is this algorithm correct?

References:

Alt, Kap. 4.8
Cormen, ch. 32 (String matching), esp. 32.4 (KMP)

Algorithmics 5

String Matching

Algorithm of Knuth-Morris-Pratt:

needs O(n) time

Implementation of main procedure (version of Iw):

```
i := 1; q := 1;
while i ≤ n do
{
    if (T[i] = P[q]) or (q = 1)
        then i := i+1
        else q := π (q-1)+1;
    if (T[i] = P[q]) then q := q+1;
    if q > m
        then
        {
            print („Matching at position “, i-m);
            q := π (q-1)+1;
        }
}
```

Invariant: q corresponds to an index such that (T[i-q+1],…,T[i-1]) coincides with (P[1],…,P[q-1])

Home work:
Why does this algorithm need O(n) time?

References:

Alt, Kap. 4.8
Cormen, ch. 32 (String matching), esp. 32.4 (KMP)

Algorithmics 5

String Matching

Algorithm of Knuth-Morris-Pratt: needs O(n) time

Implementation of prefix function (according to Cormen/Alt): needs O(m) time

```
π(1) := 0;
q := 0;
for i := 2 to m do
{
    while (P(q+1)≠P(i)) and (q > 0) do
        q := π(q);
    if P(q+1)=P(i)
         then q := q+1;
    π(i) := q
}
```

In class:
Why does this algorithm need O(m) time?

In class:
Why is this algorithm correct?

References:

Alt, Kap. 4.8
Cormen, ch. 32 (String matching), esp. 32.4 (KMP)